CLAIMS

1. An electronic document editor, comprising:

a default event handler to process editing events;

a designer extensibility mechanism to communicate with an extension coupled with the editor, the extension being configured to process at least one of the editing events; and

wherein the designer extensibility mechanism provides the editing events to the extension prior to the default event handler processing the editing events.

2. The electronic document editor as recited in claim 1, further comprising an extension interface having a pre-handle event method through which the designer extensibility mechanism provides the events to the extension.

3. The electronic document editor as recited in claim 1, wherein the designer extensibility mechanism further provides the editing events to the extension after the default event handler processes the editing events.

4. The electronic document editor as recited in claim 3, further comprising an extension interface having a pre-handle event method and a post-handle event method, the designer extensibility mechanism providing the editing events to the extension through the pre-handle event method prior to the default event handler processing the editing events, and providing the editing events to the extension through the post-handle event method after the default event handler has processed the editing events.

5. The electronic document editor as recited in claim 1, wherein:

the extension is a first extension; and

the designer extensibility mechanism further provides the editing events to a second extension after the editing events have been provided to the first extension, and prior to the default event handler processing the editing events.

6. The electronic document editor as recited in claim 5, wherein the designer extensibility mechanism further provides the editing events to the first extension and the second extension after the default event handler processes the editing events.

7. The electronic document editor as recited in claim 1, wherein:

the extension is a first extension;

the designer extensibility mechanism further provides the editing events to a second extension; and the designer extensibility mechanism further provides notice to the first extension of any action taken on an event by the second extension or the default event handler.

8. In an extensible editor having an editor extension coupled therewith, the editor having an edit designer interface comprising a pre-handle event method, the pre-handle event method comprising:

routing an editing event to the editor extension before the editor acts on the editing event; and

receiving notification from the editor extension indicating whether the editor should continue to process the editing event after the editing event has been routed to the editor extension.

9. The method as recited in claim 8, wherein the routing an editing event to the editor extension further comprises routing an event identifier to the editor extension, the event identifier uniquely identifying an editing event.

10. The method as recited in claim 8, wherein the routing an editing event to the editor extension further comprises routing an event object interface to the editor extension, the event object interface providing the editor extension with means to process the editing event.

11. In an extensible editor having an editor extension coupled therewith, the editor having an edit designer interface comprising a post-handle event method, the post-handle event method comprising:

routing an editing event to the editor extension after the editor acts on the editing event; and

receiving notification from the editor extension indicating whether the editor should continue to process the editing event after the editing event has been routed to the editor extension.

12. The method as recited in claim 11, wherein the routing an editing event to the editor extension further comprises routing an event identifier to the editor extension, the event identifier uniquely identifying an editing event.

13. The method as recited in claim 11, wherein the routing an editing event to the editor extension further comprises routing an event object interface to the editor extension, the event object interface providing the editor extension with means to process the editing event.

14. A system, comprising:

an extensible editor that processes editing events, the extensible editor having an event routing controller and a default event handler;

an extension coupled with the extensible editor for processing the editing events; and

wherein the event routing controller provides an editing event received by the editor to the extension prior to providing the editing event to be processed by the default event handler.

15. The system as recited in claim 14, wherein:

the extension is a first extension;

the system further comprises a second extension for processing the editing events; and

the event routing controller provides the editing event to the second extension prior to providing the event to the default event handler.

16. The system as recited in claim 14, wherein the event routing controller provides the editing event to the extension after the default event handler has processed the editing event.

17. The system as recited in claim 14, wherein:

the extension is a first extension;

the system further comprises a second extension for processing the editing events;

the event routing controller routes editing events to the first extension, the second extension and the default event handler; and

each event is notified of any action taken in response to the editing event by the other extension or by the default event handler.

18. The system as recited in claim 14, wherein the event routing controller further provides an event identifier uniquely identifying the editing event.

19. The system as recited in claim 14, wherein the event routing controller provides an event object interface to the extension to allow the extension to access information regarding the editing event.

20. The system as recited in claim 14, wherein the editor further comprises an edit designer interface that includes a pre-handle event method for providing the event to the extension prior to the default event handler receiving the event.

21. The system as recited in claim 14, wherein:

the editor further comprises an edit designer interface that includes a post-handle event method for providing the editing event to the extension after the default event handler has processed the event; and

the event routing controller is further configured to provide the editing event to the extension through the edit designer interface.

22. The system as recited in claim 14, wherein:

the extension is a first extension;

the editor further comprises an edit designer interface that includes a post-editor event notify method that is called by the event routing controller after the editing event has been processed by the first extension, a second extension and the default event handler to provide data to the first extension and the second extension regarding actions taken in response the editing event.

23. A designer attached to an editor, comprising a pre-event handler that processes an editing event from the editor before the editor processes the event.

24. The designer as recited in claim 23, further comprising a post-event handler that processes the editing event after the editor processes the editing event.

25. The designer as recited in claim 23, wherein the pre-event handler processes the editing event and notifies the editor to prevent further processing on the event.

26. The designer as recited in claim 23, wherein the default event handler responds to the editing event by notifying the editor to continue processing the event.

27. The designer as recited in claim 23, further comprising a post-event handler that processes the editing event after the editor processes the editing event, and notifies the editor to prevent further processing on the event.

28. The designer as recited in claim 23, further comprising a post-event handler that processes the editing event after the editor processes the event, and notifies the editor to continue processing the editing event.

29. An editor that communicates with a first designer and a second designer, comprising:

a default event handler; and

an edit designer interface that includes a pre-handle event method to process an event before the default event handler processes the event, and a post-handle event method to process the event after the default event handler has processed the event.

30. The editor as recited in claim 29, further comprising a post-editor event notify method that is called to notify the first extension of an action taken by the second extension when processing the event.

31. The designer as recited in claim 23, further comprising a translate accelerator method that translates commands received by the editor.

32. The designer as recited in claim 23, wherein the pre-handle event method and the post-handle event method include an event ID parameter that uniquely identifies the event.

33. The designer as recited in claim 23, wherein the pre-handle event method and the post-handle event method include an event object interface that allows the extensions to obtain information about the event.

34. An edit designer interface in an extensible editor, comprising:

a pre-handle event method used to send an editing event to a designer attached to the editor prior to the editor processing the editing event; and

a post-handle event method used to send the editing event to the designer after the editor has processed the editing event.

35. The edit designer interface as recited in claim 34, wherein:

the designer is a first designer;

the edit designer interface further comprises a post-editor event notify method used to notify a second designer attached to the editor of an action taken by the first designer when the first designer processed the editing event.

36. The edit designer interface as recited in claim 34, wherein the pre-handle event method and the post-handle event method include an event ID parameter that uniquely identifies the editing event.

37. The edit designer interface as recited in claim 34, wherein the pre-handle event method and the post-handle event method include an event object interface associated with the editing event through which the designer can obtain information regarding the editing event.

38. A method for processing events in an extensible editor that communicates with a first extension and a second extension, the method comprising:

sending an event to the first extension; and

receiving a signal from the first extension indicating whether to continue processing the event.

39. The method as recited in claim 38, further comprising notifying the second extension about actions taken by the first extension in response to receiving the event if the signal indicates that the event should not be processed further.

40. The method as recited in claim 38, further comprising processing the event if the signal indicates that processing should continue.

41. The method as recited in claim 38, further comprising sending the event to the second extension if the signal indicates that processing should continue.

42. The method as recited in claim 38, further comprising:

determining if the event is a command; and

if the event is a command, translating the command and withholding the event from the extensions.

43. The method as recited in claim 38, further comprising:

determining if the event is a command; and

if the event is a command, translating the command and notifying the extensions that the command has been processed.

44. A computer-readable medium having computer-executable instructions that, when executed on a computer, perform the following steps:

detect an editing event;

routing the editing event to a designer;

receiving a response from the designer that indicates whether the editing event was consumed by the first designer.

45. The computer-readable medium as recited in claim 44, further comprising computer-executable instructions to perform the following step:

processing the editing event if the response from the designer indicates that the editing event was not consumed by the first designer.

46. The computer-readable medium as recited in claim 44, wherein the designer is a first designer, and further comprising computer-executable instructions to perform the following step:

notifying a second designer that the first designer consumed the event if the response from the first designer indicates that the first designer consumed the event.

47. The computer-readable medium as recited in claim 44, wherein the designer is a first designer, and further comprising computer-executable instructions to perform the following step:

routing the event to the second designer if the response from the first designer indicates that the first designer did not consume the event.